

[Get started](#) · [Exercises](#)

Exercise 06: add a widget

🕒 2 min read · Updated 25 September 2026

Add a tiny interactive element to the wiki the way every widget is added — schema entry, custom element, fallback text, e2e test with a mocked API, budget — and see the build refuse each shortcut.

<https://docs.gurbanisoul.com/exercises/06-add-a-widget/>

Goal. Add `<sggs-clock>`: a line showing the current pahar (the three-hour watch), computed in the browser from the clock. **Needs.** the docs-site setup. 60 minutes. The widget will not be merged; the exercise is the process.

1. Try the shortcut first

Put `<!-- sggs:clock -->` in any page and run `make docs-check`. Expected: an error — unknown widget. The schema is the registry; nothing renders without an entry.

2. Declare it

Add to `docs-site/plugins/widgets.schema.json`:

```
"clock": { "purpose": "The current pahar, computed in the browser.", "attrs": {} }
```

Run `make docs-check` again. Expected: a *different* error — the widget needs static fallback text within the next 3 lines. Add a sentence under the comment (“On the rendered wiki this shows the current pahar...”). Now the gate passes, and on GitHub the page reads sensibly.

3. Write the element

Create `docs-site/src/widgets/clock.ts` — a class extending `HTMLElement` that, in `connectedCallback`, computes `Math.floor(new Date().getHours() / 3)` and writes a sentence with `textContent` (never `innerHTML`). Register it in `src/widgets/index.ts` with `define('sggs-clock', ...)`. Build and look:

```
make docs-dev
```

4. Prove it

Add a test to `docs-site/e2e/site.spec.ts`: go to your page, expect `sggs-clock` to contain “pahar”. Run `make docs-e2e`. Then check the budget:

```
cd docs-site && npm run build && npm run check:budget
```

Expected: the JS figure moves by well under a kilobyte; the ceiling is 60 KB gzipped per page.

5. Read what you did not need

Your widget made no request, so it needed no fixture. A widget that calls the API must degrade to its fallback when the API is down and must be tested with a mocked response — read `docs-site/e2e/helpers.ts` and the Ang explorer’s test to see how. Then delete your branch.

Self-check

Why must every widget be followed by a fallback sentence?

- ☐ Because the same Markdown is read on GitHub, where the widget is an invisible comment
- ☐ For search engines
- ☐ It is optional

A widget shows scripture. Where may the text come from?

- ☐ Only from the API's response, verbatim with its Ang
- ☐ From a constant in the widget, if copied carefully
- ☐ From a translation, since it is not Gurmukhi